REMARKS

Reconsideration of this application is respectfully requested.

The erroneously submitted claim 15 has now been cancelled thus obviating the Examiner's formal objections/rejections of this claim (which should never have been added to this particular application in any event).

In response to the Examiner's objection to the drawings under 37 C.F.R. §1.83(a), an attached separate Letter to the Chief Draftsperson requests permission to make further proposed drawing amendments. As will be seen in the attachment, legends are proposed so as to make it clear that <u>application</u> data is <u>transformed</u> by a dynamic proxy server <u>and</u> that different data <u>transport</u> protocols may also be used for transmitting the transformed data to the personal computer 300 (i.e., rather than the traditional HTTP application protocol/TCP transport protocol).

In view of the proposed further drawing amendments, formal drawings have not yet been submitted. However, in response to the Examiner's approval for the requested changes, a complete set of substitute formal drawings will be timely submitted.

The Examiner's provisional allowance of dependent claims 3 and 9 is appreciatively noted. No further comment will be made with respect to those provisionally allowed claims.

The rejection of claims 1, 2, 4-8, 10 and 11 under 35 U.S.C. §102 as allegedly anticipated by Katseff '796 is respectfully traversed.

With respect, it appears that the Examiner continues to confuse transport protocols with application protocols. That is, although the Examiner correctly identifies both UDP and TCP as transport protocols, the Examiner seems to equate transmission of given data by successive different transport protocols as somehow involving a "transformation" of the data being transported. However, that clearly is not the case. If the data is to be transformed, then there must be a transformation effected at the application level (rather than the lower transport level).

In particular, the passages relied upon by the Examiner to teach transformation of received data from one data format to another data format for transmission without substantially changing its information content (column 2, lines 19-51 and column 5, line 63 to column 6, line 55) deal only with converting transport protocols between TCP and UDP -- and has nothing whatever to do with transformation of the data being transmitted via the transport protocols. The passages relied upon by the Examiner do not effect any data transformation at the application level. That is, there is no transformation of data from a first encoding format to a second encoding format without substantially changing the information content of the data, etc. Rather, it is only at a lower transport level that a transport protocol is changed so as to transmit data. Although the data is admittedly not

changed (e.g., because it is not even transformed), neither is the data transformed from a first encoding format to a second encoding format.

The same deficiencies of Katseff are found with respect to independent claims 6 and 11. Dependent claims add yet further patentable distinctions.

As previously noted, Katseff discloses what appears to be a straightforward implementation of TCP and UDP in different parts of an ISP network in accordance with the known advantages and disadvantages of the network protocols (i.e., TCP for the dial-up links and UDP for the higher-capacity links within an ISP network or between different ISPs). The Examiner appears to be interpreting the conversion from TCP to UDP (or vice versa) to be transformation of data from a first coding format to a second encoding format, as re cited, for example, in claim 1.

However, UDP and TCP are both transport protocols, i.e., they are used to package data that is passed from the application layer to the transport layer (using the terminology of the OSI model). In an example given in the present application, PCM encoded audio data will be divided and encapsulated within TCP segments, which are then placed with IP packets, before the IP packets are topped and tailed with the header and trailer which are required by the bearer system, for example Ethernet. If these packets are to be converted to UDP-based packets then the reverse process will be performed (i.e., removal of the bearer header and trailer, extraction of the TCP segments from the IP packets and then removal of the encoded audio data from the TCP segments).

UDP datagrams will then be formed using the encoded audio data, which are then placed within the IP packets before bearer system header and trailer are added (see the attached extract from "TCP/IP: Architecture, protocols and implementation", S. Feit, pp 35-37, pub. McGraw-Hill). However, the conversion from TCP to UDP transport protocols does not affect the data being carried to/from the application layer and thus there is no teaching in Katseff that discloses transformation of a data encoding format without substantially changing the information content of the data.

Furthermore, there is nothing in the teaching of Katseff that would lead an ordinarily skilled person in the art to abandon conversion of the transport layer protocol, i.e., from TCP to UDP, to adopt the conversion of an encoding format at the application layer, given the complete absence from Katseff of any indication that such a change was possible or considered.

In view of the above described fundamental deficiencies of Katseff, it is not believed necessary at this time to further describe the additional deficiencies of this reference with respect to either the independent claims or the dependent claims.

Accordingly, this entire application is now believed to be in allowable form and a formal Notice to that effect is respectfully solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: Larry S. Nixon

Larry S. Nixon
Reg. No. 25,640

LSN:vc
1100 North Glebe Road, 8th Floor
Arlington, VA 22201-4714
Telephone: (703) 816-4000
Facsimile: (703) 816-4100

3.5.1 IP Actions

If the destination for a datagram is not on the same network as the source host, IP in the host directs the datagram to a local router. If that router is not connected to the destination network, the datagram must be sent on to another router. This continues until the destination network is reached.

IP makes a routing decision by looking up a remote destination in a routing table. IP locks for an entry in the routing table that matches the destination with the identity of the next router to which datagram traffic should be relayed.

3.5.2 Routing Table Information

In a small, static internet, routing tables can be entered and maintained manually. In larger internets, routers keep their tables up-to-date by exchanging information with one another. Routers can dynamically discover facts such as

- A new network has been added to the internet.
- The path to a destination has been disrupted, and that network cannot be reached at this time.
- A new router has been added to the internet. This router provides a shorter path to certain destinations.

Surprisingly, there is no single required standard for router-to-router information exchange.

The routers under the control of an organization are called an *Autonomous System*. The organization can choose any router information exchange protocol that it wants within its own Autonomous System. A router information exchange protocol used within an Autonomous System is called an *Interior Gateway Protocol* or IGP.

The *Routing Information Protocol* (RIP) is a popular Interior Gateway Protocol. One reason for the popularity of RIP is that it is so widely available. However, the newer *Open Shortest Path First* (OSPF) protocol has a rich set of useful features. The availability and popularity of OSPF are growing steadily.

Some router vendors provide their own proprietary protocols for router-to-router information exchange as well as support for standard protocols. A number of vendors have the ability to run several protocols at the same time; their routers can exchange information with other routers using any of these protocols.

3.6 TCP ARCHITECTURE

TCP is implemented in hosts. The TCP entity at each end of a connection must insure that the data delivered to its local application is

- Accurate
- In sequence
- Complete
- Free of duplicates

A sending application passes a stream of bytes to TCP. TCP breaks the stream into pieces and adds a header to each piece, forming *segments*. TCP then passes each segment to IP for transmission in a datagram (see Figure 3.6).

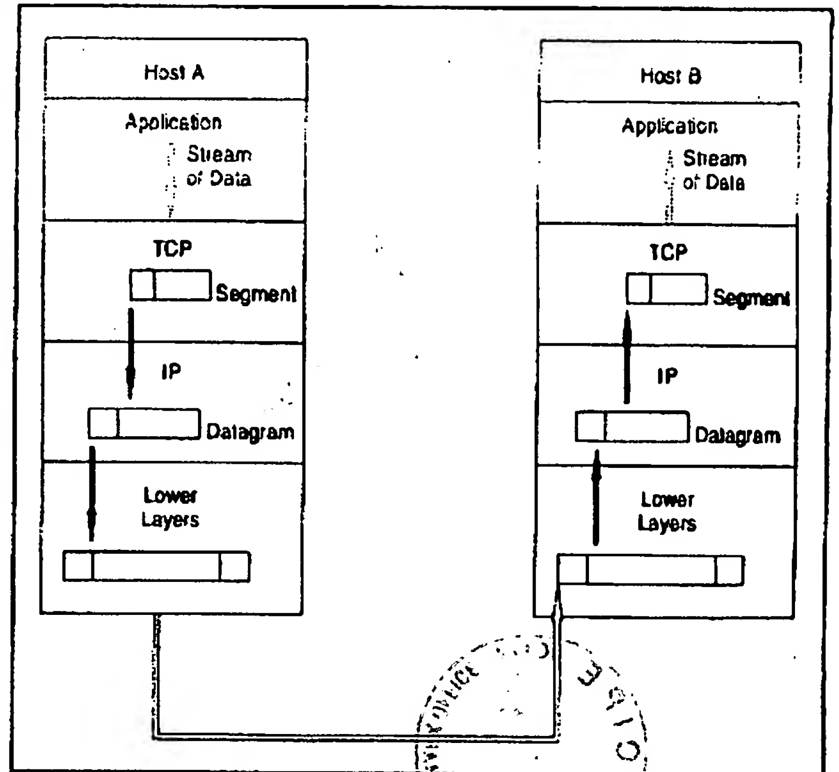

Figure 3.6 Packaging a stream of application data

A receiving TCP has to keep its partner informed of how much correct data has arrived by means of acknowledgments (ACKs). If an ACK for a segment does not arrive within a timeout period, TCP resends the segment. This strategy is called retransmission with positive acknowledgment. Occasionally, a retransmission will cause duplicate segments to be delivered to the receiving TCP.

The receiving TCP must arrange incoming segments in the right order, discarding duplicates. TCP delivers data to its application in order, without any missing pieces.

So far we have made it sound as if one side sends and the other receives. TCP is a *full duplex* protocol; that is, both ends of a connection can send and receive at the same time, so there are in fact two streams of data being transmitted. TCP can play a sender role and a receiver role simultaneously, as illustrated in Figure 3.7.

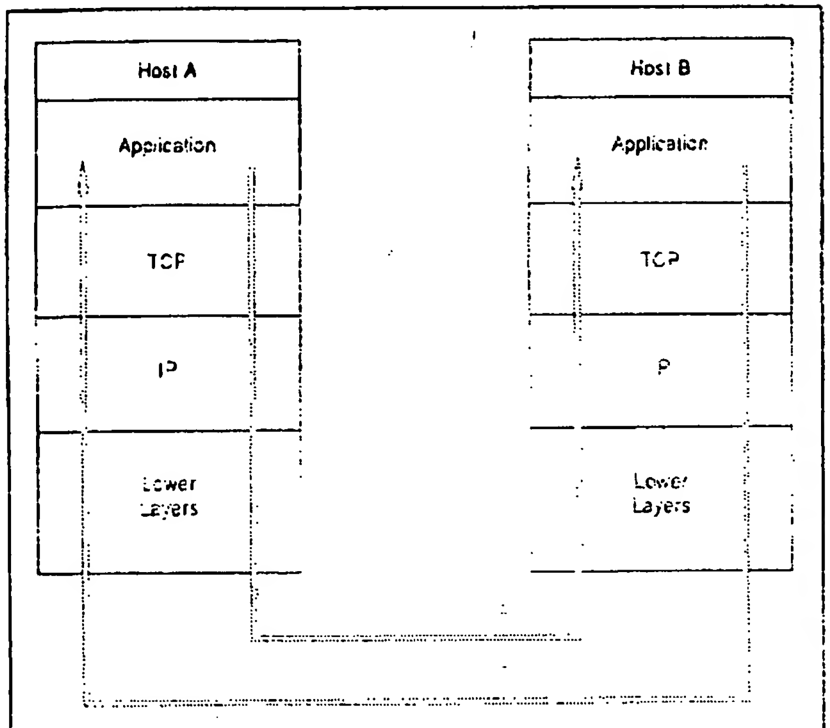

Figure 3.7 The two streams of data in a TCP connection.

3.7 UDP ARCHITECTURE

The UDP layer is implemented in host end systems. UDP makes no promise of guaranteed delivery, and it is up to the peer applications to exchange information that confirms that data has arrived safely.

An application that wants to send data via UDP passes a *block* of data to UDP. UDP adds a header to the block, forming a *User Datagram*. The User Datagram is then passed to IP and packaged in an IP datagram.

Figure 3.8 illustrates how a block of data is packaged and sent via UDP. Of course, UDP messages may be sent by either partner, and Host B could concurrently be in the process of preparing a block to be sent to A.

An application participating in UDP communications may send and receive User Datagram messages at any time. It is up to the clients and servers that are built on top of UDP to keep track of any relationship between the User Datagrams that are exchanged.

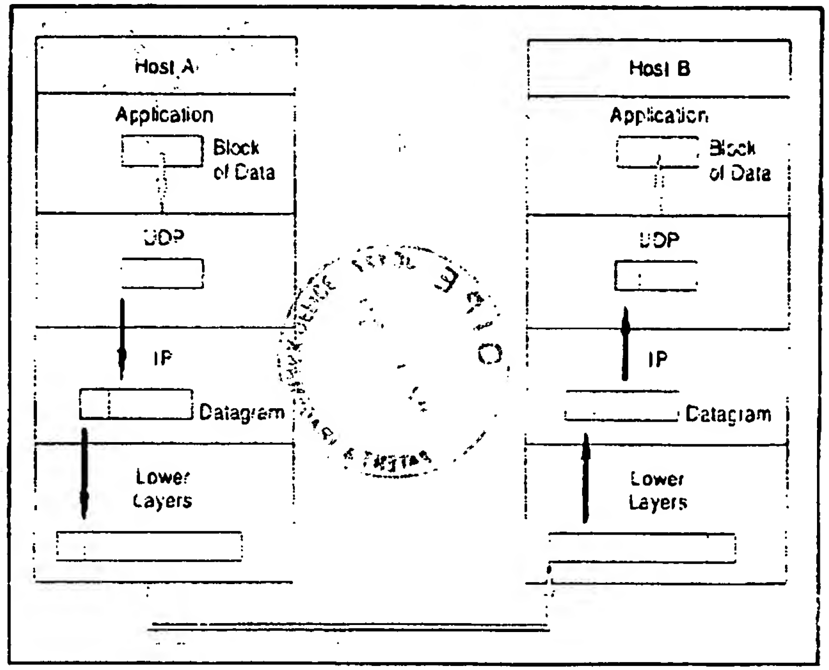

Figure 3.8 Packaging a block of application data.